ColabSomGeo: Ferramenta de monitoramento de poluição sonora subsidiada por modelo de aprendizagem

Sannyer Cardoso Carvalho Nery
carvalhosannyer@gmail.com
Instituto Federal de Educação, Ciência e Tecnologia do Amazonas
Manaus, Amazonas

Fani Tamires de Souza Batista
fani.tdsb@gmail.com
Instituto Federal de Educação, Ciência e Tecnologia do Amazonas
Manaus, Amazonas

Sergio Augusto Coelho Bezerra
sergio.bezerrar@ifam.edu.br
Instituto Federal de Educação, Ciência e Tecnologia do Amazonas
Manaus, Amazonas

ABSTRACT

Sound pollution in urban centers is a public health issue, with excessive noise negatively impacting the physical and mental health of the population. Traditional monitoring methods are often costly, making widespread data collection difficult. This paper introduces ColabSomGeo, a low-cost, scalable web tool for visualizing and analyzing georeferenced sound pollution data. Data collection is performed collaboratively through a Telegram chatbot, where users submit audio recordings and their locations. A server application processes the data, classifies the noise type using a machine learning model, and stores the results. The tool aims to assist public managers in formulating policies, support scientific research, and raise public awareness about the impacts of noise on well-being. The code is available at: https://github.com/colabsomgeo

KEYWORDS

Poluição sonora, classificação de sons urbanos, sistemas colaborativos, georreferenciamento, monitoramento, Restful, API, Chatbot

1 INTRODUÇÃO

O som é conceituado como um fenômeno constante e recorrente no cotidiano, sendo originado pela vibração mecânica. Ao ser propagado em um meio elástico, geralmente no ar, sob a forma de ondas longitudinais, é percebido pelo ouvido humano quando suas frequências estão dentro da faixa perceptível, sendo de 20Hz a 20.000Hz [8]. Por outro lado, o ruído é frenquentemente definido como um som indesejado, inoportuno ou perturbador. Embora fisicamente o ruído e som sejam semelhantes, o ruído é contextualizado pela subjetividade do ouvinte e pela função do som no ambiente. Segundo Bruel e Kjær [2], ruído é todo som que provoca incômodo ou interfere na comunicação, bem como no bem-estar humano.

Além disso, o ruído pode ser classificado em diversas categorias, como ruído contínuo, intermitente, impulsivo ou aleatório, dependendo de sua fonte e padrão temporal [4]. Os elevados níveis de decibéis (dB), resultantes de diversas atividades humanas, geram ruídos que comprometem o silêncio ambiental e o equilíbrio sonoro dos espaços urbanos [10].

De acordo com a Organização Mundial da Saúde [11], a exposição prolongada a níveis elevados de ruído está associada a problemas como estresse, distúrbios do sono, perda auditiva, hipertensão e até problemas cognitivos em crianças. Assim, a distinção entre som e ruído torna-se crucial para os estudos em acústica ambiental, engenharia, saúde e urbanismo.

Diante desse cenário, o monitoramento da poluição sonora em centros urbanos se torna essencial. Contudo, os métodos tradicionais de medição mais frequentemente usados dependem de dispositivos físicos de alto custo, o que dificulta a ampla coleta de dados [9]. Nos últimos anos, tecnologias emergentes e técnicas de Inteligência Artificial (IA) têm se consolidado como ferramentas eficazes para a análise de grandes volumes de dados em sistemas de monitoramento ambiental [1]. Nesse contexto, os sistemas colaborativos, ou de *crowdsourcing*, emergem como soluções promissoras, permitindo que cidadãos contribuam com dados por meio de seus próprios dispositivos móveis [3].

Dessa forma, este trabalho apresenta o ColabSomGeo, uma ferramenta *web* colaborativa desenvolvida para o monitoramento da poluição sonora, onde a classificação dos dados de áudio é subsidiada por um modelo de aprendizagem de máquina. A ferramenta se apresenta como uma solução de baixo custo, acessível e escalável, permitindo desta forma, o mapeamento e a análise da poluição sonora urbana. Com base no ColabSomGeo, gestores públicos poderão tomar conhecimento sobre a distribuição e os tipos de ruídos em diferentes regiões de sua cidade, permitindo desta maneira um planejamento das políticas públicas mais assertivas. Além disso, esta ferramenta pode servir de base a pesquisas científicas, inclusive que visem a promoção da saúde coletiva urbana, bem como de trabalhadores em instituições públicas e privadas.

O restante deste artigo está organizado em três seções. A Seção 1 apresenta as informações introdutórias, a Seção 2 descreve a ferramenta ColabSomGeo e a Seção 3 traz a conclusão.

2 FERRAMENTA COLABSOMGEO

Nesta seção, detalha-se a arquitetura geral da ferramenta, descrevendo cada um dos componentes desenvolvidos e suas interações. Uma demonstração do funcionamento da ferramenta proposta pode ser encontrada em https://github.com/colabsomgeo ao executar o vídeo "Vídeo Demonstração da Ferramenta". Vale ressaltar que a licença da ferramenta é gratuita somente para fins acadêmicos.

2.1 Arquitetura Geral

A Figura 1 exibe a estrutura geral da plataforma proposta, demonstrando um fluxo adequado para captura e tratamento dos dados, de tal forma que os usuários possam contribuir e visualizar.

In: XXIV Workshop de Ferramentas e Aplicações (WFA 2025). Anais Estendidos do XXXI Simpósio Brasileiro de Sistemas Multimídia e Web (WFA'2025). Rio de Janeiro/RJ, Brasil. Porto Alegre: Brazilian Computer Society, 2025.

ISSN 2596-1683

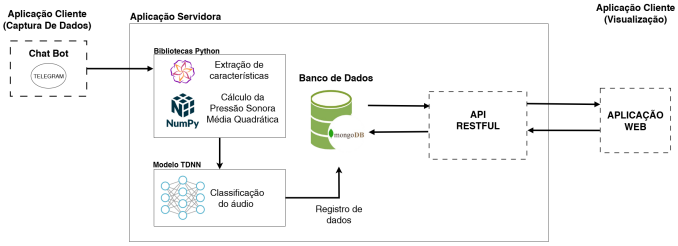

Figure 1: Macro Arquitetura da Ferramenta.

Conforme ilustrado na Figura 1, o fluxo de dados inicia-se na Aplicação Cliente de Captura de Dados, na qual um *Chatbot*, implementado na plataforma Telegram atua como interface de interação com o usuário. Vale ressaltar que um *Chatbot* oferece uma forma de coleta de dados com baixo atrito, uma vez que não exige a instalação de um novo aplicativo. Por meio desse recurso, é realizada a coleta de dados de áudio e informações de localização de maneira acessível.

Uma vez recebido, os dados são direcionados à Aplicação Servidora, que concentra o processamento e a camada de inteligência da ferramenta. Nesta etapa, um conjunto de Bibliotecas Python é empregado para a extração de informações relevantes, incluindo a extração de características acústicas do áudio e o cálculo da pressão sonora média quadrática, utilizando a biblioteca Numpy [5], para as operações numéricas.

O áudio capturado inicialmente no formato .ogg é convertido para .wav, compatível com o modelo de classificação adotado. Os atributos extraídos e os resultados da classificação são então armazenados no MongoDB Atlas, banco de dados NoSQL escolhido pela sua escalabilidade e flexibilidade no gerenciamento de dados não estruturados.

Para disponibilizar os dados de forma acessível, foi desenvolvida uma API (*Application Programing Interface*) que atua como ponte entre o banco de dados e a Aplicação Cliente de Visualização. Esta última consiste em uma aplicação *Web* que consome os recursos da API e apresenta os resultados em formato de mapas de calor, completando o ciclo desde a captura até a visualização das informações.

2.2 Aplicação Cliente de Captura de Dados

A porta de entrada para a contribuição de dados no sistema é a Aplicação Cliente de Captura de Dados, composta por um chatbot desenvolvido para a plataforma de mensagens Telegram. A escolha dessa plataforma mostrou-se estratégica, uma vez que combina facilidade de uso com a robustez de sua API, a qual permite a automação de interações e o envio de diferentes tipos de mídia, incluindo arquivos de áudio e dados de gerenciamento. Dessa forma, a adesão do Telegram foi utilizada apenas para os testes com a ferramenta, havendo a possibilidade de outros tipos de entradas de dados, como WhatsApp e um aplicativo que poderá ser desenvolvido especificamente para a captura de dados.

O principal objetivo desse componente é fornecer uma interface simples e acessível para o envio de amostras de áudio e informações de geolocalização. O fluxo de interação, ilustrado na Figura 2, foi concebido de modo a ser intuitivo:

(1) Início da Interação: O usuário localiza o bot no Telegram e inicia uma conversa. O bot responde com uma mensagem de boas-vindas e fornece instruções claras sobre como gravar e enviar uma amostra de áudio.
(2) Envio de Localização e Áudio: O usuário envia a localização e realiza a gravação de uma mensagem de voz diretamente na interface do chat.
(3) Confirmação e Feedback: Após o envio, o bot captura o arquivo de áudio e os metadados associados (como ID do usuário e data/hora). Em seguida, envia uma mensagem de confirmação ao usuário, informando que os dados foram recebidos com sucesso e encaminhados para análise.

Figure 2: Interação entre o usuário e o *Chat Bot* no Telegram.

2.3 Aplicação Servidora

A aplicação Servidora é o núcleo da ferramenta, responsável por orquestrar a interação com usuário, processar os dados recebidos, armazená-los de forma estruturada e disponibilizá-los para consulta. A aplicação é implementada em Python e pode ser dividida em três etapas principais: recepção e gerenciamento de dados; pré-processamento e análise do áudio; e persistência dos dados.

O ponto de entrada da aplicação é o *script* principal, que inicializa o bot Telegram e configura um *ConversationHandler*. Este componente da biblioteca python-telegram-bot é responsável por gerenciar o diálogo com o usuário como uma máquina de estados, garantindo que a coleta de dados siga uma sequência lógica.

O fluxo de conversação é iniciado com o comando */start*. Nesse momento, o sistema verifica se o usuário já contribuiu anteriormente, por meio da função acharUser (*user_id*) para consultar o banco de dados. Caso seja um novo usuário, seus dados básicos (ID do telegram e nome) são cadastrados através da função salvarUser (*user_id, first_name*). Concluída esta etapa, o *bot* solicita ao usuário que envie a localização geográfica, passando para o estado *LOCATION*.

Ao receber a localização, a função *handle_location* é acionada, armazenando as coordenadas de latitude, longitude, data e hora da captura, então solicita o envio de uma amostra de áudio, mudando a conversa para o estado de *AUDIO*. Após o envio da mensagem de áudio pelo usuário, a função *handle_audio* assume o controle. O áudio, é recebido no formato .ogg e baixado para o servidor onde se encontra a aplicação. Em seguida, a biblioteca *torchaudio* é utilizada para converter o arquivo para o formato .wav, que é mais adequado para as etapas subsequentes. Com o arquivo no formato correto, duas funções principais, pertencentes ao módulo de processamento, são executadas, a saber:

(1) A função extrair_informacoes_audio(caminho_arquivo) é chamada para analisar o áudio. Utilizando a biblioteca librosa, ela calcula métricas acústicas importantes, como o nível de pressão sonora mínimo e máximo (em dB), a média quadrática (RMS) que indica a intensidade do som, e aplica um filtro de ponderação A (A-Weighting) para simular a percepção do ouvido humano, gerando valores em dBA.
(2) Classificação do Áudio: Em seguida, a função classificar_audio utiliza um modelo de classificação de áudio pré-treinado da biblioteca SpeechBrain Urbansound8k_ECAPA. Este modelo analisa o arquivo .wav e retorna a categoria de som mais provável. Para garantir a qualidade dos dados, a classificação só é aceita se o modelo apresentar um nível de confiança de no mínimo 60%. Caso contrário, o áudio é rotulado como "outros".

A escolha do SpeechBrain se deu por ser um *toolkit* de código aberto baseado em PyTorch [6], que se destaca pela facilidade de combinação de difrentes módulos, para criação de modelos complexos [7]. A arquitetura ECAPA-TDNN (*Emphasized Channel Attention, Propagation and Aggregation in TDNN*) é aplicada sobre o conjunto de dados UrbanSound8k, resultando no modelo específico utilizado neste trabalho, UrbanSound8K_ecapa. Essa combinação é empregada para classificar eventos sonoros urbanos de forma eficaz.

O UrbanSound8K é um *dataset* amplamente utilizado para o treinamento e avaliação de classificadores de som ambiental. Ele é composto por 8.732 arquivos de áudio classificados em dez categorias distintas, contendo sons de ambientes urbanos, como buzina, latido de cachorro, furadeira, motor e sirene. Para a classificação, os áudios do *dataset* são primeiramente convertidos em espectrogramas ou utilizados diretamente como forma de onda para alimentar o modelo ECAPA. Este modelo extrai representações discriminativas para treinar a rede classificadora [12].

Para a validação das classificações, foi empregado uma taxa de confiança de 60%. A escolha desse limiar representa um equilíbrio entre a precisão, a proporção de classificações corretas entre as predições positivas, e o recall, a capacidade do modelo de identificar todas as amostras relevantes. Um limiar mais alto aumentaria a precisão, todavia poderia descartar classificações corretas com menor confiança. Desse modo, quando uma predição do modelo atinge ou ultrapassa 60% de confiança, é considerada valida; caso contrário, o áudio é classificado como pertencente à categoria "Outros" para otimizar a captura de eventos sonoros relevantes sem introduzir um número excessivo de falsos positivos.

Após a extração e classificação, as informações coletadas são agrupadas em um documento que, por sua vez, é salvo no banco de dados por meio da função salvarInfoAudio (dadosAudio). Esta função possibilita a conexão de instância MongoDB onde é inserido o registro na coleção *captura_som*. O módulo mongoDB.py gerencia toda a comunicação com o banco de dados, incluindo a conexão, autenticação e as operações de escrita e leitura nas coleções.

Cada documento salvo na coleção representa uma única gravação de áudio enviada por um usuário. Os atributos armazenados fornecem um contexto completo sobre a amostra, incluindo quem a enviou, onde e quando foi gravada, suas propriedades acústicas e o que o sistema identificou no som. A seguir, são listados os nomes e as descrições dos atributos:

- **`id_user_object`:** Este é o identificador único que vincula a gravação ao usuário que a enviou.
- **latitude e longitude:** Estes campos armazenam as coordenadas geográficas exatas de onde o áudio foi capturado.
- **data_criacao:** Corresponde à data e hora exatas em que a captura foi iniciada pelo usuário.
- **min_db e max_db:** Representam o nível de pressão sonora mínimo e máximo do áudio, medido em decibéis (dB).
- **min_dBA e max_dBA:** Semelhante aos valores em dB, mas com a aplicação de um filtro de ponderação A (dBA).
- **rms_loudness:** O valor da Média Quadrática (Root Mean Square).
- **categoria:** Esta é a etiqueta de classificação atribuída ao áudio pelo modelo de inteligência artificial.

Após a conclusão bem sucedida do salvamento, os arquivos de áudio temporários (.ogg e .wav) são removidos do servidor para liberar espaço. O bot, por fim, envia uma mensagem de confirmação ao usuário, finalizando o ciclo de captura e processamento. Para disponibilizar os dados armazenados no MongoDB de forma segura e padronizada, foi desenvolvida uma API Restful utilizando a Framework FastAPI em Python. Esta API atua como a ponte entre o banco de dados e a aplicação cliente de visualização. Para garantir a comunicação com a interface de visualização, que pode estar hospedada em um domínio diferente, foi implementado um *middleware* CORS (CORSMiddleware), permitindo que a aplicação web faça requisição à API de forma segura.

O principal ponto de acesso aos dados é o **endpoint**, *GET* /api/audios, que é responsável por consultar a coleção e retornar uma lista de registros de áudio. Para refinar os resultados, este *endpoint* aceita um parâmetro de consulta opcional chamado categoria. Se uma categoria for especificada na URL, a API realiza uma busca filtrada, retornando apenas os documentos que correspondem à categoria solicitada. Caso nenhum parâmetro seja fornecido, todos os registros da coleção são retornados.

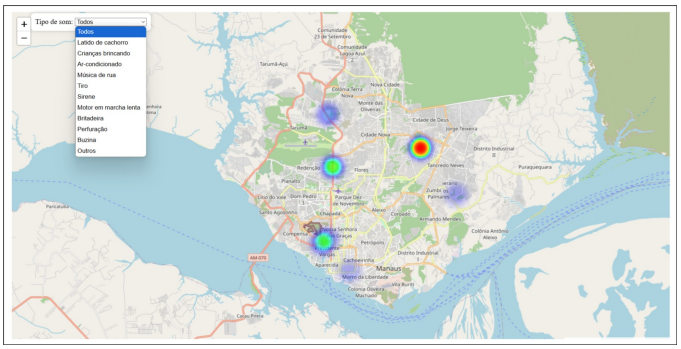

Figure 3: Página web da Ferramenta.

2.4 Aplicação Cliente de Visualização

A Aplicação cliente de visualização é uma página web interativa, construída com HTML, CSS e JavaScript, projetada para permitir a exploração geográfica dos dados de poluição sonora. A interface consiste em um mapa de calor interativo, renderizado pela biblioteca Leaft.js. Os dados sonoros são plotados como um mapa de calor. Nesta visualização, os pontos representam as localizações das gravações, e a intensidade do calor é proporcional ao nível máximo de ruído medido (max_dBA), oferecendo uma percepção visual imediata das áreas com maior impacto sonoro.

Para permitir uma análise mais detalhada, a página oferece um mecanismo de filtragem por meio de um menu suspenso. Este componente permite ao usuário selecionar uma categoria específica de som, como "Sirene", "Buzina" ou "Música de rua", para isolar e visualizar apenas os eventos sonoros de interesse no mapa. A página web é ilustrada na Figura 3.

3 CONCLUSÃO

O sistema desenvolvido demonstrou a viabilidade de uma solução colaborativa baseada em Aprendizado de Máquina para a classificação e o monitoramento georreferenciado de ruídos urbanos. Contudo, os resultados preliminares também evidenciaram a necessidade de ampliar os experimentos com um conjunto de dados mais diversificado, de modo a aumentar a acurácia e a capacidade de generalização do modelo.

Dessa forma, em trabalhos futuros, serão incorporadas melhorias com o objetivo de ampliar a usabilidade do sistema por colaboradores e gestores públicos. No que se refere à acurácia da coleta, projeta-se o desenvolvimento de um aplicativo especializado. Diferentemente das plataformas de mensagens, geralmente otimizadas para a faixa de frequência da voz humana, um software dedicado possibilitaria a captura de um espectro sonoro mais amplo e detalha-do. Além disso, a implementação de rotinas de calibração, por meio da comparação entre os dados obtidos por *smartphones* e aqueles registrados por medidores de nível de pressão sonora devidamente certificados, cons-titui uma etapa essencial para a correção de desvios e o aumento da confiabilidade das medições.

Para endereçar o desafio de ruídos espúrios e potenciais submissões maliciosas, inerentes a sistemas colaborativos, futuras iterações da ferramenta preveem a implementação de estratégias de mitigação. Em uma frente, serão exploradas abordagens estatísticas para detecção de anomalias, analisando a consistências espacial e temporal dos dados para identificar registros que divirjam significativamente do padrão local. Adicionalmente, planeja-se o desenvolvimento de um sistema de validação por pares, no qual colaboradores com maior nível de engajamento ou reputação possam auditar e confirmar a veracidade de amostras enviadas por outros usuários, fortalecendo a confiabilidade e a integridade da base de dados.

AGRADECIMENTOS

Os autores agradecem às Instituições: Fundação de Apoio ao Ensino, Pesquisa, Extensão e Interiorização (Faepi), Instituto Federal de Educação, Ciência e Tecnologia do Amazonas (IFAM) e seu Curso Superior em Análise e Desenvolvimento de Sistemas (TADS). E ao projeto Aranouá financiado pela Samsung Eletrônica da Amazônia.

REFERENCES

[1] C. A. R. Beltran et al. 2019. Plataforma de aprendizado de máquina para detecção e monitoramento de alunos com risco de evasão. (2019), 1591.
[2] Brüel & Kjær. 1996. Environmental Noise. Technical Review, No. 3.
[3] Ana Mercês Nicolaci da Costa and Mariano Pimentel. 2011. Sistemas colaborativos para uma nova sociedade e um novo ser humano. In *Sistemas Colaborativos* (1 ed.), Mariano Pimentel and Hugo Fuks (Eds.). Vol. 1. Elsevier, Rio de Janeiro, 3–15.
[4] Samir N. Y. Gerges. 2000. *Ruído: Fundamentos e Controle* (2 ed.). NR Editora, Florianópolis.
[5] Numpy Developers. [n. d.]. NumPy. https://numpy.org/. Acesso em: 19 de agosto de 2023.
[6] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., 8024–8035.
[7] Mirco Ravanelli, Titouan Parcollet, Peter Plantinga, Aku Rouhe, Samuele Cornell, Loren Lugosch, Cem Subakan, ASR Gradient, Hwidong Na, Vimal Panayotov, et al. 2021. SpeechBrain: A general-purpose speech toolkit. In *arXiv preprint arXiv:2106.04624*.
[8] Thomas D. Rossing, F. Richard Moore, and Paul A. Wheeler. 2014. *The Science of Sound* (3 ed.). Pearson, San Francisco.
[9] A. S. Santos et al. 2020. Desafios e oportunidades da aplicação de Sistemas Ciberfísicos no monitoramento da poluição urbana. (2020).
[10] Igor Sabarense Coelho Silva and Felipe Cunha. 2022. Monitoramento da Poluição Sonora em Belo Horizonte utilizando Redes de Sensoriamento Participativo. Trabalho acadêmico.
[11] World Health Organization. 2018. *Environmental Noise Guidelines for the European Region.* Technical Report. WHO Regional Office for Europe, Copenhagen. https://www.who.int/publications/i/item/9789289053563
[12] Binyu Zhao and Jian Liang. 2024. Hierarchical-Concatenate Fusion TDNN for sound event classification. *Plos one* 19, 10 (2024), e0312998.